Package ‘botor’

March 12, 2023

Type Package
Title 'AWS Python SDK' ('boto3') for R
Description Fork-safe, raw access to the 'Amazon Web Services' ('AWS') 'SDK' via the 'boto3' 'Python' module, and convenient helper functions to query the 'Simple Storage Service' ('S3') and 'Key Management Service' ('KMS'), partial support for 'IAM', the 'Systems Manager Parameter Store' and 'Secrets Manager'.
SystemRequirements Python and boto3 (https://aws.amazon.com/sdk-for-python)
Version 0.4.0
Date 2023-03-12
URL https://daroczig.github.io/botor/
BugReports https://github.com/daroczig/botor/issues
RoxygenNote 7.1.1
License AGPL-3
Encoding UTF-8
Imports utils, reticulate, checkmate, logger, jsonlite
Suggests testthat, covr, digest
NeedsCompilation no
Author Gergely Daróczzi [aut, cre] (<https://orcid.org/0000-0003-3149-8537>), System1 [fnd]
Maintainer Gergely Daróczzi <daroczig@rapporter.net>
Repository CRAN
Date/Publication 2023-03-12 15:00:02 UTC

R topics documented:

  boto3 ......................................................... 2
  boto3_version ............................................... 3
  botor ....................................................... 3
boto3

Raw access to the boto3 module imported at package load time

Description

Raw access to the boto3 module imported at package load time

Usage

boto3
**Format**

An object of class python.builtin.module (inherits from python.builtin.object) of length 0.

**Note**

You may rather want to use `botor` instead, that provides a fork-safe boto3 session.
Arguments

aws_access_key_id
AWS access key ID

aws_secret_access_key
AWS secret access key

aws_session_token
AWS temporary session token

region_name
Default region when creating new connections

botocore_session
Use this Botocore session instead of creating a new default one

profile_name
The name of a profile to use. If not given, then the default profile is used

Value

boto3 Session

botor_client
Creates an initial or reinitialize an already existing AWS client or resource cached in the package's namespace

Description

Creates an initial or reinitialize an already existing AWS client or resource cached in the package's namespace

Usage

botor_client(service, type = c("client", "resource"), cache = TRUE, ...)

Arguments

service
string, eg S3 or IAM

type
AWS service client or resource to be created, eg s3

cache
boolan flag for caching the client or resource in the package namespace. For (internal) package functions, it's best to set to TRUE to avoid reinitializing the client/resource, but for custom use and when you need to use multiple clients for the same service in parallel (eg working with different regions etc), you might want to set this to FALSE

... further parameters passed to the client or resource, eg endpoint_url

Value

cached AWS client

References

check_s3_uri

Check if an argument looks like an S3 bucket

Description
Check if an argument looks like an S3 bucket

Usage

check_s3_uri(x)

Arguments

x
string, URI of an S3 object, should start with s3://, then bucket name and object key

Examples

check_s3_uri('s3://foo/bar')
check_s3_uri('https://foo/bar')
## Not run:
assert_s3_uri('https://foo/bar')
## End(Not run)

iam
The default, fork-safe IAM client on the top of boto3

Description
The default, fork-safe IAM client on the top of boto3

Usage

iam()

Value
boto3.client.IAM

References
iam_get_user

Description
Retrieves information about the specified IAM user, including the user’s creation date, path, unique ID, and ARN.

Usage
iam_get_user(...) 

Arguments
...  

Value
list

References
https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/iam.html#IAM.Client.get_user

iam_whoami

Description
Get the current AWS username

Usage
iam_whoami()

Value
string

See Also
sts_whoami
kinesis

The default, fork-safe Kinesis client on the top of boto3

Description
The default, fork-safe Kinesis client on the top of boto3

Usage
kinesis()

Value
botocore.client.Kinesis

References

kinesis_describe_stream

Describes the specified Kinesis data stream

Description
Describes the specified Kinesis data stream

Usage
kinesis_describe_stream(stream)

Arguments
stream the name of the stream to describe

Value
list

References
kinesis_get_records  
*Gets data records from a Kinesis data stream’s shard*

**Description**

Gets data records from a Kinesis data stream’s shard

**Usage**

```
kinesis_get_records(shard_iterator, limit = 25L)
```

**Arguments**

- `shard_iterator`  
  the position in the shard from which you want to start sequentially reading data records, usually provided by `kinesis_get_shard_iterator`
- `limit`  
  maximum number of records to return

**Value**

list of Records, NextShardIterator and MillisBehindLatest

**References**


**Examples**

```r
## Not run:
botor(profile_name = 'boto3-tester')
iterator <- kinesis_get_shard_iterator(stream = 'boto3-tester', shard = '0')
kinesis_get_records(iterator$ShardIterator)

## End(Not run)
```

---

kinesis_get_shard_iterator  
*Gets an Amazon Kinesis shard iterator*

**Description**

Gets an Amazon Kinesis shard iterator
**kinesis_put_record**

**Usage**

```r
kinesis_put_record(
    stream, data, partition_key, ...
)
```

**Arguments**

- `stream` the name of the stream to describe
- `shard` the shard ID of the Kinesis Data Streams shard to get the iterator for
- `shard_iterator_type` determines how the shard iterator is used to start reading data records from the shard
- `...` optional further parameters, such as StartingSequenceNumber or Timestamp

**Value**

- list of ShardIterator

**References**


**See Also**

- `kinesis_get_records`

---

**kinesis_put_record**  
*Writes a single data record into an Amazon Kinesis data stream*

**Description**

Writes a single data record into an Amazon Kinesis data stream

**Usage**

```r
kinesis_put_record(stream, data, partition_key, ...)
```
Arguments

- **stream**: the name of the stream to describe
- **data**: the data blob (<1 MB) to put into the record, which is base64-encoded when the blob is serialized
- **partition_key**: Unicode string with a maximum length limit of 256 characters determining which shard in the stream the data record is assigned to
- ... optional further parameters, such as ExplicitHashKey or SequenceNumberForOrdering

Value

list of ShardId, SequenceNumber and EncryptionType

References


---

**kms**

*The default, fork-safe KMS client on the top of boto3*

Description

The default, fork-safe KMS client on the top of boto3

Usage

```
kms()
```

Value

`botocore.client.KMS`

References

**kms_decrypt**

*Decrypt cipher into plain text via KMS*

**Description**

Decrypt cipher into plain text via KMS

**Usage**

```
kms_decrypt(cipher, simplify = TRUE)
```

**Arguments**

- `cipher` Base64-encoded ciphertext
- `simplify` returns decrypted plain-text instead of raw list

**Value**

decrypted text as string or list

**See Also**

`kms_encrypt`

**kms_decrypt_file**

*Decrypt file via KMS*

**Description**

Decrypt file via KMS

**Usage**

```
kms_decrypt_file(file, return = file)
```

**Arguments**

- `file` base file path (without the enc or key suffix)
- `return` where to place the encrypted file (defaults to file)

**Value**

decrypted file path

**See Also**

`kms_encrypt kms_encrypt_file`
**kms_encrypt**

Encrypt plain text via KMS

**Usage**

```r
kms_encrypt(key, text, simplify = TRUE)
```

**Arguments**

- `key` the KMS customer master key identifier as a fully specified Amazon Resource Name (eg `arn:aws:kms:us-east-1:123456789012:key/12345678-1234-1234-1234-123456789012`) or an alias with the alias/ prefix (eg `alias/foobar`)
- `text` max 4096 bytes long string, eg an RSA key, a database password, or other sensitive customer information
- `simplify` returns Base64-encoded text instead of raw list

**Value**

string or list

**See Also**

- `kms_decrypt`

**kms_encrypt_file**

Encrypt file via KMS

**Usage**

```r
kms_encrypt_file(key, file)
```

**Arguments**

- `key` the KMS customer master key identifier as a fully specified Amazon Resource Name (eg `arn:aws:kms:us-east-1:123456789012:key/12345678-1234-1234-1234-123456789012`) or an alias with the alias/ prefix (eg `alias/foobar`)
- `file` file path
kms_generate_data_key

Value
two files created with enc (encrypted data) and key (encrypted key) extensions

See Also
kms_encrypt kms_decrypt_file

---

kms_generate_data_key  Generate a data encryption key for envelope encryption via KMS

Description
Generate a data encryption key for envelope encryption via KMS

Usage
kms_generate_data_key(key, bytes = 64L)

Arguments
- **key**: the KMS customer master key identifier as a fully specified Amazon Resource Name (eg `arn:aws:kms:us-east-1:123456789012:key/12345678-1234-1234-1234-123456789012`) or an alias with the alias/ prefix (eg `alias/foobar`)
- **bytes**: the required length of the data encryption key in bytes (so provide eg 64L for a 512-bit key)

Value
list of the Base64-encoded encrypted version of the data encryption key (to be stored on disk), the raw object of the encryption key and the KMS customer master key used to generate this object

---

mime_guess  Guess the type of a file based on the filename using mimetypes Python module

Description
Guess the type of a file based on the filename using mimetypes Python module

Usage
mime_guess(file)

Arguments
- **file**: path
s3

The default, fork-safe Amazon Simple Storage Service (S3) client on the top of boto3

Usage

s3()

Value

s3.ServiceResource

References


s3_copy

Copy an object from one S3 location to another

Description

Copy an object from one S3 location to another

Usage

s3_copy(uri_source, uri_target)

Arguments

uri_source  string, location of the source file
uri_target  string, location of the target file

Value

invisibly uri_target

References

**s3_delete**

Delete an object stored in S3

**Usage**

```r
s3_delete(uri)
```

**Arguments**

- `uri` string, URI of an S3 object, should start with `s3://`, then bucket name and object key

---

**s3_download_file**

Download a file from S3

**Usage**

```r
s3_download_file(uri, file, force = TRUE)
```

**Arguments**

- `uri` string, URI of an S3 object, should start with `s3://`, then bucket name and object key
- `file` string, location of local file
- `force` boolean, overwrite local file if exists

**Value**

invisibly file

**References**

Examples

```r
## Not run:
s3_download_file('s3://botor/example-data/mtcars.csv', tempfile())

## End(Not run)
```

---

`s3_exists` Checks if an object exists in S3

**Description**
Checks if an object exists in S3

**Usage**

```r
s3_exists(uri)
```

**Arguments**

- **uri** string, URI of an S3 object, should start with `s3://`, then bucket name and object key

**Value**

boolean

**Examples**

```r
## Not run:
s3_exists('s3://botor/example-data/mtcars.csv')
s3_exists('s3://botor/example-data/UNDEFINED.CSVLX')

## End(Not run)
```

---

`s3_list_buckets` List all S3 buckets

**Description**
List all S3 buckets

**Usage**

```r
s3_list_buckets(simplify = TRUE)
```
**s3_ls**

*List objects at an S3 path*

**Arguments**

- `simplify` return bucket names as a character vector

**Value**

- list of `boto3.resources.factory.s3.Bucket` or a character vector

**s3_ls**

List objects at an S3 path

**Usage**

`s3_ls(uri)`

**Arguments**

- `uri` string, should start with `s3://`, then bucket name and optional object key prefix

**Value**

- data.frame with `bucket_name`, `object_key`, `uri` (that can be directly passed to eg `s3_read`), `size` in bytes, `owner` and `last_modified` timestamp

**s3_object**

*Create an S3 Object reference from an URI*

**Description**

Create an S3 Object reference from an URI

**Usage**

`s3_object(uri)`

**Arguments**

- `uri` string, URI of an S3 object, should start with `s3://`, then bucket name and object key

**Value**

`s3$Object`
**s3_put_object_tagging**  
Sets tags on s3 object overwriting all existing tags. Note: tags and metadata tags are not the same

**Description**
Sets tags on s3 object overwriting all existing tags. Note: tags and metadata tags are not the same

**Usage**
```
s3_put_object_tagging(uri, tags)
```

**Arguments**
- **uri**  
  string, URI of an S3 object, should start with s3://, then bucket name and object key
- **tags**  
  named character vector, e.g. `c(my_first_name = 'my_first_value', my_second_name = 'my_second_value')` where names are the tag names and values are the tag values.

**References**

**s3_read**  
Download and read a file from S3, then clean up

**Description**
Download and read a file from S3, then clean up

**Usage**
```
s3_read(uri, fun, ..., extract = c("none", "gzip", "bzip2", "xz"))
```

**Arguments**
- **uri**  
  string, URI of an S3 object, should start with s3://, then bucket name and object key
- **fun**  
  R function to read the file, eg fromJSON, stream_in, fread or readRDS
- **...**  
  optional params passed to `fun`
- **extract**  
  optionally extract/decompress the file after downloading from S3 but before passing to `fun`
s3_upload_file

Value

R object

Examples

## Not run:
s3_read('s3://botor/example-data/mtcars.csv', read.csv)
s3_read('s3://botor/example-data/mtcars.csv', data.table::fread)
s3_read('s3://botor/example-data/mtcars.csv2', read.csv2)
s3_read('s3://botor/example-data/mtcars.RDS', readRDS)
s3_read('s3://botor/example-data/mtcars.json', jsonlite::fromJSON)
s3_read('s3://botor/example-data/mtcars.jsonl', jsonlite::stream_in)

## read compressed data
s3_read('s3://botor/example-data/mtcars.csv.gz', read.csv, extract = 'gzip')
s3_read('s3://botor/example-data/mtcars.csv.gz', data.table::fread, extract = 'gzip')
s3_read('s3://botor/example-data/mtcars.csv.bz2', read.csv, extract = 'bzip2')
s3_read('s3://botor/example-data/mtcars.csv.xz', read.csv, extract = 'xz')

## End(Not run)

s3_upload_file

Upload a file to S3

Description

Upload a file to S3

Usage

s3_upload_file(file, uri, content_type = mime_guess(file))

Arguments

file string, location of local file
uri string, URI of an S3 object, should start with s3://, then bucket name and object key
content_type content type of a file that is auto-guess if omitted

Value

invisibly uri

References

s3_write

Write an R object into S3

Description
Write an R object into S3

Usage
s3_write(x, fun, uri, compress = c("none", "gzip", "bzip2", "xz"), ...)

Arguments
- x: R object
- fun: R function with file argument to serialize x to disk before uploading, eg write.csv, write_json, stream_out or saveRDS
- uri: string, URI of an S3 object, should start with s3://, then bucket name and object key
- compress: optionally compress the file before uploading to S3. If compression is used, it’s better to include the related file extension in uri as well (that is not done automatically).
- ...: optional further arguments passed to fun

Note
The temp file used for this operation is automatically removed.

Examples
---
## Not run:
t <- tempfile()
write.csv(mtcars, t, row.names = FALSE)
s3_upload_file(t, 's3://botor/example-data/mtcars.csv')
unlink(t)
## note that s3_write would have been a much nicer solution for the above
## End(Not run)
Examples

```r
## Not run:
s3_write(mtcars, write.csv, 's3://botor/example-data/mtcars.csv', row.names = FALSE)
s3_write(mtcars, write.csv2, 's3://botor/example-data/mtcars.csv2', row.names = FALSE)
s3_write(mtcars, jsonlite::write_json, 's3://botor/example-data/mtcars.json', row.names = FALSE)
s3_write(mtcars, jsonlite::stream_out, 's3://botor/example-data/mtcars.jsonl', row.names = FALSE)
s3_write(mtcars, saveRDS, 's3://botor/example-data/mtcars.RDS')

## compress file after writing to disk but before uploading to S3
s3_write(mtcars, write.csv, 's3://botor/example-data/mtcars.csv.gz',
         compress = 'gzip', row.names = FALSE)
s3_write(mtcars, write.csv, 's3://botor/example-data/mtcars.csv.bz2',
         compress = 'bzip2', row.names = FALSE)
s3_write(mtcars, write.csv, 's3://botor/example-data/mtcars.csv.xz',
         compress = 'xz', row.names = FALSE)

## End(Not run)
```

### sm

The default, fork-safe AWS Systems Manager (SecretManager) client on the top of botor

#### Description

The default, fork-safe AWS Systems Manager (SecretManager) client on the top of botor

#### Usage

```r
sm()
```

#### Value

`botocore.client.secretsmanager`

#### References

sm_get_secret  
*Read AWS System Manager’s Secrets Manager via Secret Manager*

**Description**
Read AWS System Manager’s Secrets Manager via Secret Manager

**Usage**
```
sm_get_secret(path, key = NULL, parse_json = TRUE)
```

**Arguments**
- `path`  
  name/path of the key to be read
- `key`  
  single key or a vector of keys.
- `parse_json`  
  logical. Default TRUE

**Value**
(optionally decrypted) value

---

**ssm**  
*The default, fork-safe AWS Systems Manager (SSM) client on the top of botor*

**Description**
The default, fork-safe AWS Systems Manager (SSM) client on the top of botor

**Usage**
```
ssm()
```

**Value**
```
botocore.client.SSM
```

**References**
ssm_get_parameter  

Read AWS System Manager’s Parameter Store

Description

Read AWS System Manager’s Parameter Store

Usage

ssm_get_parameter(path, decrypt = TRUE)

Arguments

- **path**: name/path of the key to be read
- **decrypt**: decrypt the value or return the raw ciphertext

Value

- (optionally decrypted) value

sts_whoami  

Returns details about the IAM user or role whose credentials are used to call the operation

Description

Returns details about the IAM user or role whose credentials are used to call the operation

Usage

sts_whoami()

Value

- list with UserId, Account and Arn

References

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/sts.html#STS.Client.get_caller_identity

See Also

iam_whoami
Index

* datasets
  boto3, 2
  assert_s3_uri(check_s3_uri), 5
  boto3, 2
  boto3_version, 3
  botor, 3, 3
  botor_client, 4
  check_s3_uri, 5
  expect_s3_uri(check_s3_uri), 5
  iam, 5
  iam_get_user, 6
  iam_whoami, 6, 23
  kinesis, 7
  kinesis_describe_stream, 7
  kinesis_get_records, 8, 9
  kinesis_get_shard_iterator, 8, 8
  kinesis_put_record, 9
  kms, 10
  kms_decrypt, 11, 12
  kms_decrypt_file, 11, 13
  kms_encrypt, 11, 12, 13
  kms_encrypt_file, 11, 12
  kms_generate_data_key, 13
  mime_guess, 13
  s3, 14
  s3_copy, 14
  s3_delete, 15
  s3_download_file, 15, 20
  s3_exists, 16
  s3_list_buckets, 16
  s3_ls, 17
  s3_object, 17
  s3_put_object_tagging, 18
  s3_read, 17, 18
  s3_upload_file, 19
  s3_write, 20
  sm, 21
  sm_get_secret, 22
  ssm, 22
  ssm_get_parameter, 23
  sts_whoami, 6, 23
  test_s3_uri(check_s3_uri), 5